# BCConversion™ Technical Manual

*Edition 2.0*

*November 2008*

For assistance, contact:
Adaptable Controllers, LLC
17212 Niwot Place
Parker, Colorado, 80134
USA
+1-303-378-4577
sales@AdaptableControllers.com
support@AdaptableControllers.com
http://www.AdaptableControllers.com/products/BCConversion

© 2001 - 2008 RenaiSys, LLC
All rights reserved.
Printed in the United States of America.

Development under GNU's General Public License
Facilitated by Adaptable Controllers LLC

Thank you for the contributions from (in alphabetical order):
**Aviosoft**
**RenaiSys LLC**
**SoftMap Inc**
**and others**

BCConversion is a replacement for the original BC electronics with it's own real-time processor.

Copyright (C) 2001 - 2008 RenaiSys LLC

RenaiSys LLC
17212 Niwot Place
Parker, CO 80134
USA

The real-time program is free software; you may redistribute it and/or modify it under the terms of the GNU General Public License version 2, as published by the Free Software Foundation.

The real-time program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to:

Free Software Foundation, Inc.
59 Temple Place - Suite 330
Boston, MA 02111-1307
USA

See Appendix A for the complete text of the license.

# Table of Contents

1. Overview....4
2. Vision....5
3. Design....6
3.1. Functional Design....6
3.2. Real-Time Software Design....7
3.3. Main Program Design....9
3.4. Firmware Design....24
3.5. Software Usage and Configuration....29
3.6. Hardware Design....32
3.6.1. Connector Configurations....33
4. Implementation....34
4.1. Software Installation....34
4.1.1. Jetway 625EMP800 Motherboard Setup....34
4.1.2. Jetway 626EMP1G Motherboard Setup....35
4.1.3. Intel D201GLY Motherboard Setup....35
4.2.1. Linux Software Installation on Hard Disk Drive....36
4.2.2. Linux Software Installation on USB Drive....37
4.2.3. Windows Software Installation on USB Drive....38
4.3. Firmware Installation....39
4.4. Hardware....40
5. Updating, Testing and Saving....41
Appendix A; Copy of GNU's General Public License....42
Appendix D; CD Directory and File Names....46
Appendix H; Document History....51
Appendix N; Networking....52
Appendix P; Firmware Programming Cable....53
Appendix R; Resources....54
Appendix Z; Credits....57

## 1. Overview

The BCConversion replaces the proprietary electronics of BC analytical plotters with an adaptable real-time processor under an open source license at a reasonable cost. Since most instruments were originally developed in the late 1970s and 1980s, some of the electronics hardware has become difficult to maintain and extremely expensive, when compared with current technology. With the availability of open-source hard real-time software and the increased processing power of new and inexpensive computer hardware, it has become possible to create a low-cost, generic interface for the plotters.

Potential benefits:

- For system owners:
    - Reduced maintenance costs
    - Extended system life (or improved resale value)
    - Broader selection of application programs
- For hardware vendors:
    - Improved sales of new systems due to good customer service
    - Reduced maintenance and development costs
    - Reduce spare part inventory costs
- For application software vendors:
    - Improved sales due to a broader customer base
    - Reduced maintenance and development costs (one version can run on many different system types)
    - Increased revenues from selling BCConversions (hardware, software, and installation services)
- For technical support organizations:
    - Improved sales of new systems due to good customer service and broader customer base
    - Reduced maintenance costs due to the open availability of source and documentation
    - Reduced travel and spare part inventory costs

The BCConversion consists of four components; application interface, real-time control software, firmware, and hardware. The open application interface allows application programs to connect to a plotter without consideration for make or model of the hardware. The real-time control software computes the motions, so the image can be seen in 3D. The firmware is software that is programmed into micro-controllers to keep close control of the hardware. The hardware is adaptable to different plotters and connects to encoders, motors, lights, switches, and displays.

The Hardware is available from Adaptable Controllers. Software, installation, integration, and support services are available from RenaiSys and other authorized service providers. The development for the BCConvertion is facilitated by Adaptable Controllers under GNU's General Public License. Anybody interested, is encouraged to contribute to the best of their abilities and provide feedback to Adaptable Controllers. We hope, manufacturers, application developers, and users of the plotters will benefit.

## 2. Vision

Technology gave rise to the Photogrammetry Industry and continues to drive the individuals as well as the organizations that are involved. In order to extract scientific information from images and process the data, photogrammetry organizations have been using analog systems, analytical systems, and digital systems, as well as radar and satellite technology. The rapid development of computer technology has opened many new doors for the industry.

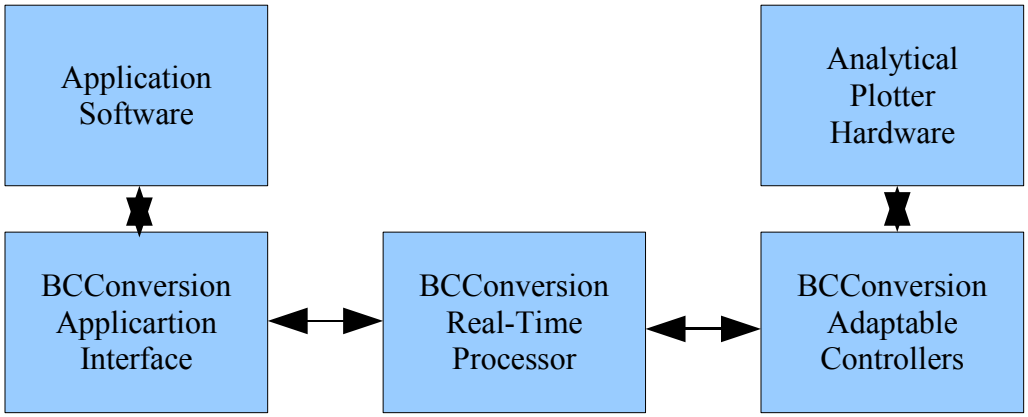
While researchers continue to push the development of the technology, organizations in production environments tend to use the most productive equipment to get the work done. As significant equipment costs are amortized, asset management becomes a challenge, particularly as newer, less expensive equipment become available.

Hardware continues to be replaced by more adaptable software technology. We see researchers openly sharing elementary findings, the international society coordinating and establishing standards, hardware and software vendors implementing new technology and further research, system owners using the technology and funding education.

At RenaiSys, we provide independent services to the industry. With the BCConversion, we hope to consolidate old, mature technology that has been proprietary or abandoned and make it widely available and easy to maintain.

## 3. Design

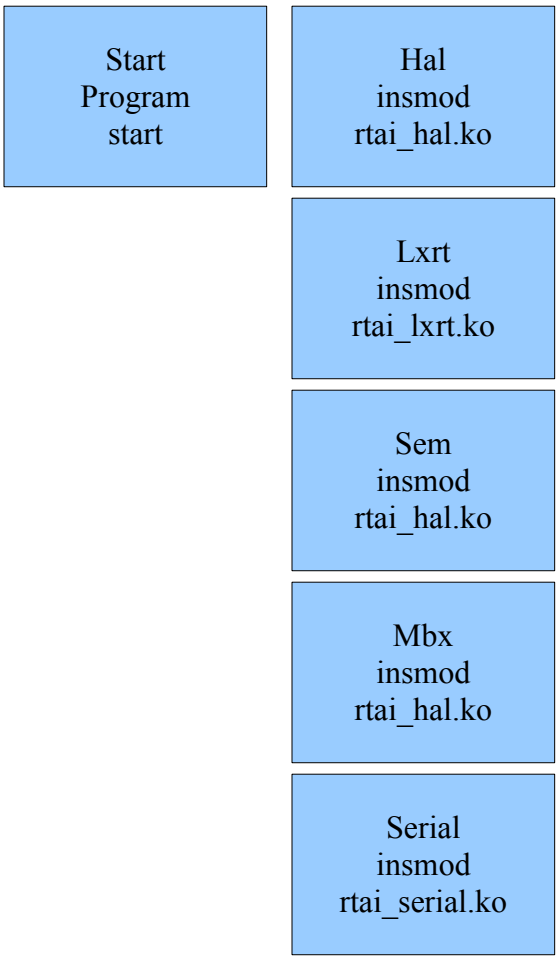
### 3.1. Functional Design



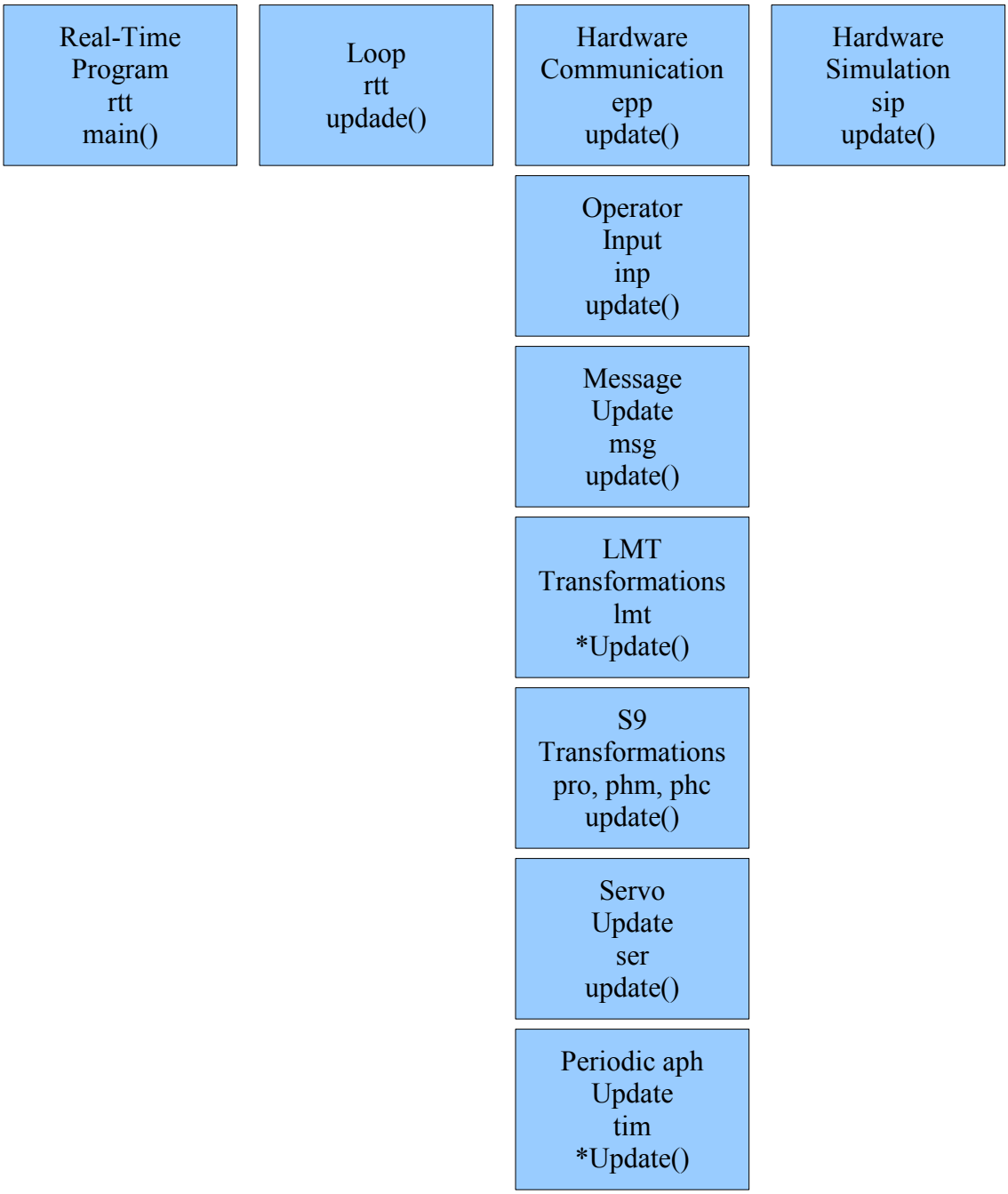

Application software like orientation and data collection software for triangulation, cross-section, volume, industrial, medical, or topographical data communicates with the BCConversion's application interface via serial (RS232) interface. The BCConversion's real-time processor communicates with the BCConversion's adaptable controllers via parallel (EPP) interface. The controllers monitor and interact with the plotter's hardware (encoders, switches, displays, and motors). The controllers communicate with each-other via controller area network (CAN).

The real-time processor's start script (/usr/bin/start called via /etc/init.d/rcS and /a/auto.sh) loads the real-time kernel modules and starts several independent threads at different priority levels. The the real-time thread (main function in src/rtt/rtt.c) with the communication interface to the Adaptable Controllers, runs as a periodic process with a higher priority than any LINUX process. The main user thread (main function in src/aph/aph.c) with the application interface runs as a regular LINUX process. In addition, there might be optional threads for keyboard input (main function in src/kbd/kbd.c), mouse input (main function in src/mou/mou.c), different simulation and monitoring threads (in src/sim and src/sip) or touch screen input (main function in src/tou/tou.c). The all threads communicate through mailboxes.

### 3.2. Real-Time Software Design



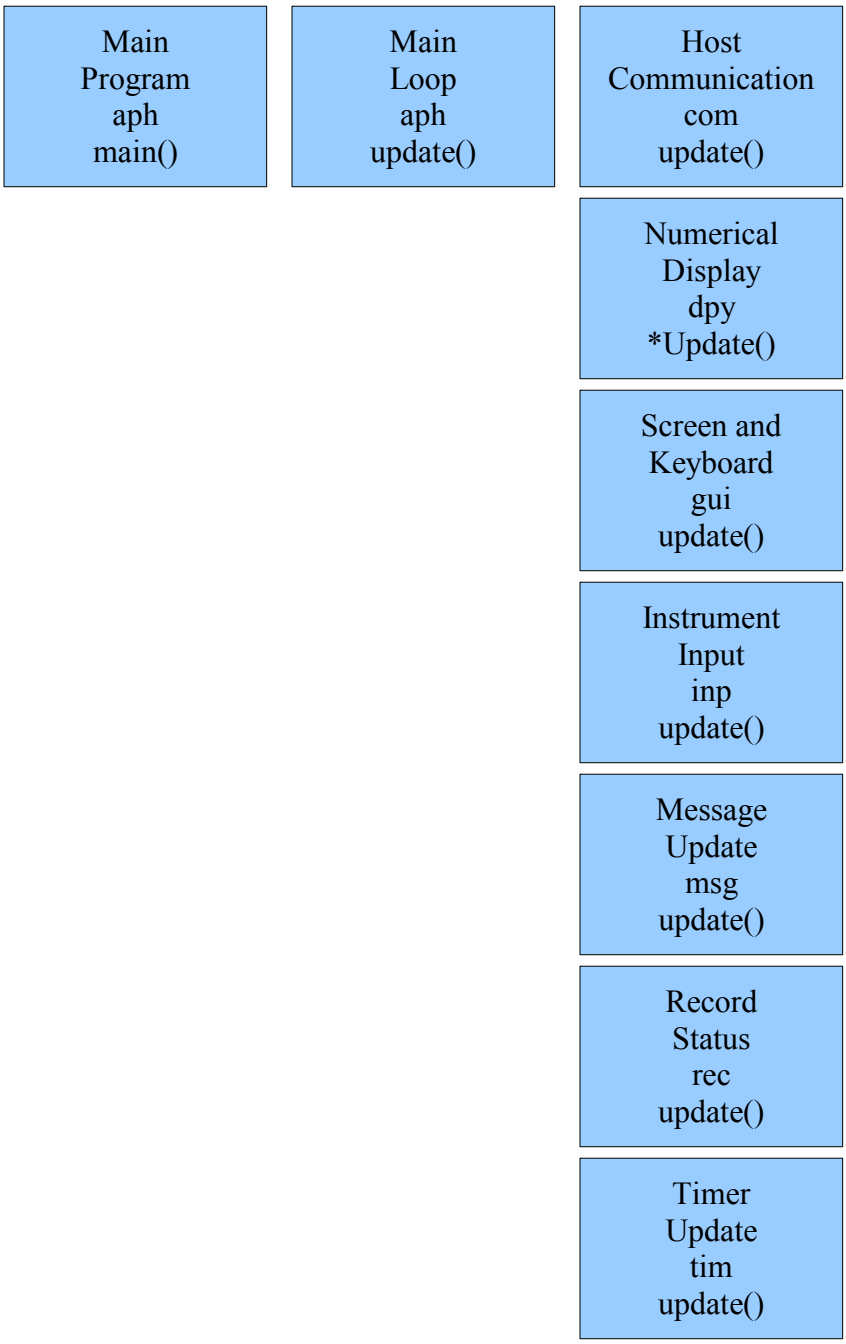

Since the LINUX kernel is patched with a modification to insert modules that receive timer and other interrupts before LINUX does, these modules have the absolute highest priority. Task scheduling as well as mailbox and serial communication buffers are managed by the RTAI modules. See scripts/start and RTAI documentation at www.rtai.org for more detail.




At a lower priority, the real-time thread communicates with the adaptable controllers. Operator input and messages from the real-time processor are considered for the transformations in the different operating modes. LMT and S9 transformations are only computed if needed. Since the thread runs as an endless loop, the sequence of the main functions should not matter. If time is available within the fix period and additional messages might get processed by calling epp update() and msg update() more than once within a single update period.

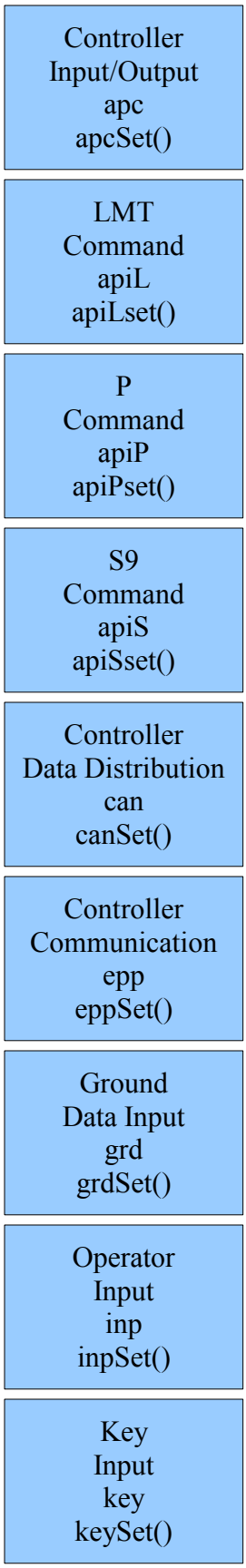
As indicated in the diagram, the main real-time modules are rtt, epp, inp, msg, lmt, phc, phm, pro, ser, and tim. In addition, there is can and sip. The can module buffers messages to be sent to the controllers and forwards messages from the controllers. The sim module has the real-time functions to simulate the hardware. It acts as sub module to epp.

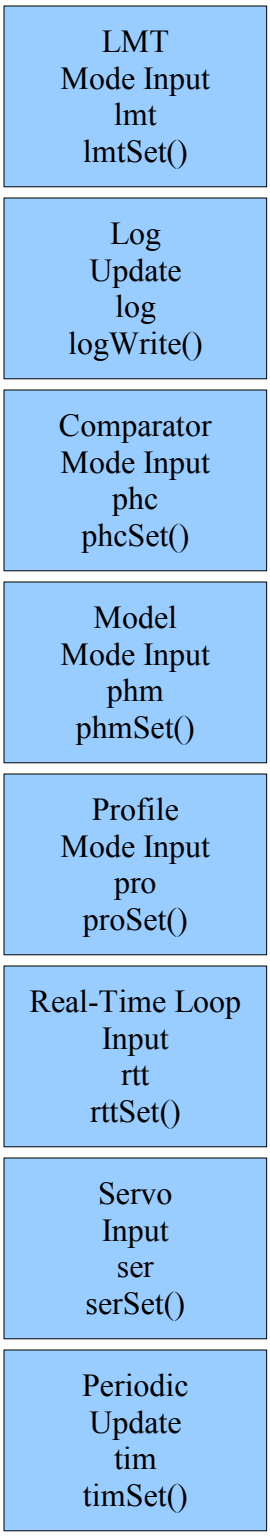
### 3.3. Main Program Design




At the lowest priority (the same as other user applications), the aph thread runs as fast as possible. It does the parsing of the application commands and updates the display.

Since the thread runs as an endless loop, the sequence of the main functions (as indicated and run in alphabetical order) should not matter. The following modules are idle. Only if a message is received (from the real-time thread, the application host, the user interface or any simulation program) are their functions called.


Controller
Input/Output
apc
apcSet()
LMT
Command
apiL
apiLset()
P
Command
apiP
apiPset()
S9
Command
apiS
apiSset()
Controller
Data Distribution
can
canSet()
Controller
Communication
epp
eppSet()
Ground
Data Input
grd
grdSet()
Operator
Input
inp
inpSet()
Key
Input
key
keySet()




As indicated in the diagram, the main aph modules are apc, apiL, apiP, apiS, can, com, dpy, epp, grd, gui, inp, key, lmt, log, msg, phc, phm, pro, rec, rtt, ser, and tim. In addition, there is cfg, the module to read and write configuration parameters to disk; sim, the module to simulate an application host; str, the module to provide the text strings; and tes, the module to test some of the aph data, arguments and functions.

```
Software flow:

aph/main()
      aph/head()
            printf()
      aph/init()
      aph/arguments()
            aph/argumentHelp()
                  printf()
            exit()
      logOpen()
            fopen()
            setlinebuf()
            time()
            snprintf()
            logWrite()
                  fprintf()
            aphErrorSet()
                  snprintf()
                  logWrite() ...
                  rt_get_adr()
                  rt_mbx_receive_wp()
      rttGet(RTT_BOX_CHECK)
            fopen()
            nam2num()
            rt_get_adr()
            snprintf()
            logWrite()
            fclose()
      logClose()
            time()
            snprintf()
            logWrite() ...
            fflush()
            fclose()
      exit()
      aph/intInit()
            signal()
      aph/refInit()
            snprintf()
            logWrite() ...
            apcGet()
            apiLGet()
            canGet()
            comGet()
            dpyGet()
            grdGet()
            guiGet()
            inpGet()
            keyGet()
            lmtGet()
            phcGet()
            phmGet()
            proGet()
            recGet()
            rttGet()
            serGet()
            timGet()
      tes/test()
            tes/arguments()
            tes/str()
            tes/canMsg()
```

```
      tes/msg()
      tes/acError()
      tes/funGet()
      tes/funSet()
      tes/funMax()
      tes/cfg()
      tes/cfgTable()
mlockall()

while(no error or exit)
      aphErrorSet() ...
      aph/modInit()
            aph/stateSet(STA_INITIALIZING)
            aph/modAllRun()
                  apc/init()
                  apiL/init()
                        snprintf()
                        logWrite() ...
                  apiS/init()
                        snprintf()
                        logWrite() ...
                  can/init()
                  cfg/init()
                  com/init()
                        fopen()
                  dpy/init()
                  epp/init()
                  grd/init()
                  gui/init()
                        printf()
                        endwin()
                        initscr()
                        start_color()
                        init_pair()
                        noecho()
                        nonl()
                        cbreak()
                        timeout()
                        keypad()
                        guiSet()
                              guiGet0()
                              snprintf()
                              aphErrorSet() ...
                              curs_set()
                              erase()
                              bkgd()
                              gui/title()
                              gui0/data()
                              gui/helpS()
                              refresh()
                              touchLoad()
                  inp/init()
                  key/init()
                  lmt/init()
                  phc/init()
                  phm/init()
                  pro/init()
                  rec/init()
                  rtt/init()
                        snprintf()
                        logWrite() ...
                        aphErrorSet() ...
```

```
            rt_mbx_init()
            rt_task_init()
            MsgToRtt()
      ser/init()
      tim/init()
            tim/fopen()
            tim/fclose()
            tim/fprintf()
            snprintf()
            aphErrorSet() ...

aph/stateSet(STA_CONFIGURING)
aph/modAllRun()
      cfg/cRead()
      lmt/config()
            lmt/offsetRead()
                  fopen()
                  snpringf()
                  logWrite() …
                  fclose()
                  sscanf()
            lmt/servoRead()
                  fopen()
                  snpringf()
                  logWrite() …
                  fclose()
                  sscanf()
                  MsgToRtt()
            lmt/keyRead()
                  fopen()
                  snpringf()
                  logWrite() …
                  fclose()
                  sscanf()
                  fread()
                  strcmp()
                  strcpy()


aph/stateSet(STA_OPENING)
aph/modAllRun()
      com/open()
            rt_spclose()
            rt_spopen()
            snprintf()
            logWrite() ...
      lmt/open
            lmt/fileRead()
                  strcpy()
                  strcat()
                  snprintf()
                  logWrite() …
                  fopen()
                  fread()
                  rewind()
                  fclose()

aph/stateSet(STA_MINI_RUN)
aph/modAllRun()
      gui/mUpdate()
            gui/titleUpdate()
            gui0/update()
```

```
            refresh()
      msg/update()

aph/stateSet(STA_RT_LOAD)
aph/modAllRun()
      phc/load()
            MsgToRtt()
      phm/load()
            phm/correctionLoad()
                  fabs()
                  MsgToRtt()
            phm/displayUpdate()
                  apcSet()
                        apc/ioUpdate()
            phm/thisLoad()
                  MsgToRtt()
            phm/inputLoad()
                  MsgToRtt()
            phm/leftSetupLoad()
                  MsgToRtt()
            phm/rightSetup/Load()
                  MsgToRtt()
            MsgToRtt()
      pro/load()
            snprinf()
            logWrite()
            MsgToRtt()


aph/stateSet(STA_HARDWARE_LOAD)
aph/modAllRun()
      apc/hLoad()
            timCheck()
                  snprintf()
                  logWrite() ...
                  aphSet(APH_RUN)
                  aphErrorSet() ...
            canApcSet()
                  snprintf()
                  logWrite() ...
                  can/check()
                  MsgToRtt()
      can/load()
            timCheck() ...
            canApcSet() ...
      com/load()
            timCheck() ...
            canApcSet() ...
      dpy/hLoad()
            timCheck() ...
            canApcSet() ...
      epp/load()
            timCheck() ...
            canApcSet() ...
      key/load()
            timCheck() ...
            keySet()
            canApcSet() ...
      ser/hLoad()
            timCheck() ...
            canApcSet() ...
```

```
      aph/stateSet(STA_INITIAL_LOAD)
      aph/modAllRun()
            apc/iLoad()
                  canApcSet() ...
            apiS/load()
            dpy/iLoad()
                  dpy/dataUpdate()
                        sprintf()
                        isdigit()
                        canApcSet() ...
            inp/load()
                  inp/factorLoad()
                        MsgToRtt
                  inp/fastLoad
                        inp/fastAdjust()
                              snprintf()
                              aphErrorSet()
                        MsgToRtt
                  inpSet()
                        apcSet()
                              apc/ioUpdate()
                                    snprintf()
                                    aphErrorSet() ...
                                    canApcSet() ...
                        inp/flagLoad()
                              MsgToRtt()
                  inp/rotationLoad()
                        MsgToRtt()
            ser/iLoad()
                  ser/apcConfig()
                        aphErrorSet() ...
                        canApcSet()
                  ser/kineticsLoad()
                        MsgToRtt()
                  ser/leftCalibrationLoad()
                        MsgToRtt()
                  ser/rightCalibrationLoad()
                        MsgToRtt()
      aph/runTableSet()
            snprintf()
            logWrite() ...
      rttSet()
            snprintf()
            aphErrorSet() ...
            dpySet()
                  dpy/ledUpdate()
                        apcSet()
                              apc/ioUpdate()
                              snprintf()
                              aphErrorSet() ...
                              canApcSet() ...
            MsgToRtt()              ##### Initialization is done


aph/update()
      poll()
      snprintf()
      logWrite() ...
      aphSet()
      aph/runTableSet()
            snprintf()
```

```
      logWrite()
aph/stateSet(STA_RUNNING)
      com/update()
            rt_spread()
            com/bufferFull()
                  snprintf()
                  aphErrorSet() ...
            com/streamLog()
                  snprintf()
                  logWrite() ...
                  sprintf()
                  fprintf()
                  fflush()
                  rt_get_adr()
                  MsgToSim()
            comSet()
            apiSetL()
                  comSet()
                        snprinf()
                        logWrite() ...
                        fprintf()
                        fflush()
                        rt_spwrite()
                        rt_get_adr()
                        MsgToSim()
                  snprintf()
                  aphErrorSet()
                  apiL/api*
                        lmtSet()
            snprintf()
            aphErrorSet() ...
            streamIndexReset()
            isprint()
            logWrite() ...
            fprintf()
            fflush()
            rt_get_adr()
            MsgToSim()
            apiSetS()
                  isspace()
                  snprinf()
                  aphErrorSet()...
                  apiS/api*
      dpy/lUpdate()
            dpy/pointUpdate()
                  dpy/dataUpdate() ...
            dpy/point3dUpdate()
                  dpy/dataUpdate() ...
            dpy/dataUpdate()
      dpy/sUpdate()
            dpy/pointUpdate()
                  dpy/dataUpdate() ...
            dpy/point3dUpdate()
                  dpy/dataUpdate() ...
            dpy/dataUpdate()
      inp/update()
            inp/leverUpdate()
                  apcGet()
            MsgToRtt()
      lmt/update()
            lmt/statusUpdate()
            lmt/recordUpdate()
```

```
                  sqrt()
                  lmt/recordLoad()
            lmt/recordProfile()
                  lmt/recordLoad()
            lmt/recordSend()
                  lmt/record3d()
                        lmt/recordLong()
                        lmt/recordCode()
                  lmt/record2d()
                        lmt/recordLong()
                        lmt/recordCode()
                  lmt/recordCheckSend()
                        comSet()
                              snpringf()
                              logWrite()
                              sprintf()
                              fprintf()
                              fflush()
                              rt_spwrite()
                              rt_get_adr()
                              MsgToSim()
                  lmt/recordSave()
                        strncpy()
      msg/update()
            rt_get_adr()
            snprintf()
            logWrite() ...
            aphErrorSet() ...
            rt_mbx_evdrp()
            canSet()
            comSet()
            inpSet()
            msgSet()
            rttSet()
            timSet()
            sprintf()
            rt_mbx_receive_wp()
      rec/update()
            apiSetS()
                  apiS/apiCoordinateSend()
                        sprintf()
                        apiS/apiAsciiFromFloat()
                        snprintf()
                        aphErrorSet() ...
                        comSet()
                              snprintf()
                              logWrite() ...
                              fprintf()
                              fflush()
                              rt_spwrite()
                              rt_get_adr()
                              MsgToSim()
            snprintf()
            aphErrorSet() ...
      tim/update()
            lmtSet()
            canApcSet()
            rt_get_adr()
            rt_mbx_receive_if()
            snprintf()
            aphErrorSet() ...
            fopen()
```

```
                        fprintf()
                        fclose()


      aph/modClose()
            aph/stateSet(STA_CLOSING)
            aph/modAllRun()
                  cfg/cWrite()
                  com/end()
            aph/stateSet(STA_MINI_WAIT)
            aph/modAllRun()

            aph/stateSet(STA_WAITING)
            aph/modAllRun()

            aph/stateSet(STA_EXITING)
            aph/modAllRun()
                  gui/end()
                        endwin()
                  rtt/end()
      sleep()
                                        #### End of loop
end()
      aph/intClose()
            signal()
      logClose() ...
exit()
```

```
Real-time thread flow:

rtt/main()
      rtt/init()
      rtt/head()
      rtt/arguments()
            getopt()
            atoi()
            rtt/agumentHelp()
                  printf()
            exit()
      rt_get_adr()
      exit()
      rtt/intInt()
            signal()
      rtt/refInit()
            canGet()
            eppGet()
            inpGet()
            lmtGet()
            phcGet()
            phmGet()
            proGet()
            serGet()
            timGet()
      rtt/rttInit()
            printf()
            nam2num()
            rt_mbx_init()
            fprintf()
            rttErrorSet()
                  MsgToAph()
                  snprintf()
                  rt_get_adr()
                  rt_mbx_receive_wp()
            rt_task_init_schmod()
            rt_set_periodic_mode()
            start_rt_timer()
            mlockall()
            rt_make_hard_real_time()
            rt_task_use_fpu()
            rt_get_time()
            rt_task_make_periodic()
            rt_make_soft_real_time()
      rtt/update()
            canGet()
            eppGet()
            msgGet()
            serGet()
            rt_get_cpu_time_ns()
            rt_task_wait_period()
            rtt/stateSet()
            rtt/runTableSet()

            #### run module functions
            rtt/stateSet(STA_INITIALIZING)
                  can/init()
                  epp/init()
                  inp/init()
                  lmt/init()
                  msg/init()
                  pro/init()
```

```
      phm/init()
      phc/init()
      ser/init()
      tim/init()

rtt/stateSet(STA_CONFIGURING)
rtt/stateSet(STA_OPENING)
rtt/stateSet(STA_MINI_RUN)
rtt/stateSet(STA_RT_LOAD)
rtt/stateSet(STA_HARDWARE_LOAD)
rtt/stateSet(STA_INITIAL_LOAD)
rtt/stateSet(STA_RUNNING)
      epp/update()
            rt_get_cpu_time_ns()
            rt_get_adr()
            MsgToSip()
            canSet()
                  serSet()
                  inpSet()
                  MsgToAph()
            outb()
            snprintf()
            MsgToAph()
            epp/readyCheck()
                  outb()
                  snprintf()
                  MsgToAph()
            inb()
      msg/update()
            rt_get_adr()
            rt_mbx_evdrp()
            canSet() ...
            eppSet() ...
            inpSet() ...
            lmtSet() ...
            phcSet() ...
            phmSet() ...
            proSet() ...
            rttSet() ...
            serSet() ...
            timSet() ...
            snprintf()
            MsgToAph()
            rt_mbx_receive_wp()
            rttErrorSet() ...
      tim/idleUpdate()
            MsgToAph()

rtt/stateSet(STA_LMT_COMPARATOR_RUNNING)
      epp/update() ...
      inp/update()
            snprintf()
            MsgToAph()
      lmt/comparatorUpdate()
            lmt/locationUpdate()
                  lmt/photoGridOneUpdate()
                  lmt/cameraOneUpdate()
                        sqrt()
                        fabs()
                  lmt/refractionOneUpdate()
                  lmt/photoToModel()
                  lmt/curvatureUpdate()
```

```
      msg/update() ...
      ser/update()
            ser/checkAll()
                  ser/check()
            ser/outAll()
                  ser/out()
                  canSet()
                        snprintf()
                        MsgToAph()
            abs()
            ser/profileStart()
            ser/profile()
      tim/lmtUpdate()
            MsgToAph()

rtt/stateSet(STA_LMT_MODEL_RUNNING)
      epp/update() ...
      inp/update() ...
      lmt/modelUpdate()
            lmt/modelToPhotoUpdate()
            lmt/cameraOneUpdate()
                  sqrt()
                  fabs()
            lmt/photoGridOneUpdate()
      msg/update() ...
      ser/update() ...
      tim/lmtUpdate()
            MsgToAph()

rtt/stateSet(STA_LMT_PROFILE_RUNNING)
      epp/update() ...
      inp/update() ...
      lmt/profileUpdate()
      msg/update() ...
      ser/update() ...
      tim/lmtUpdate()
            MsgToAph()

rtt/stateSet(STA_P_COMPARATOR_RUNNING)
      epp/update() ...
      inp/update() ...
      msg/update() ...
      ser/update() ...

rtt/stateSet(STA_P_MODEL_RUNNING
      epp/update() ...
      inp/update() ...
      msg/update() ...
      ser/update() ...

rtt/stateSet(STA_P_PROFILE_RUNNING)
      epp/update() ...
      inp/update() ...
      msg/update() ...
      ser/update() ...

rtt/stateSet(STA_S9_COMPARATOR_RUNNING)
      epp/update() ...
      inp/update() ...
      msg/update() ...
      phc/update()
      ser/update() ...
```

```
            tim/comparatorUpdate()
                  MsgToAph()

      rtt/stateSet(STA_S9_MODEL_RUNNING)
            epp/update() ...
            inp/update() ...
            msg/update() ...
            phm/update()
                  phm/comparatorFromModel()
                        phm/correctionApply()
            phc/modelUpdate()
            ser/update() ...
            tim/modelUpdate()
                  MsgToAph()

      rtt/stateSet(STA_S9_PROFILE_RUNNING)
            epp/update() ...
            inp/update() ...
            msg/update() ...
            pro/update()
                  rttSet()
            phm/profileUpdate()
                  phm/comparatorFromModel()
                        phm/correctionApply()
            phc/modelUpdate()
            ser/update() ...
            tim/profileUpdate()
                  MsgToAph()

      rtt/stateSet(STA_IDLE)
            epp/update() ...
            msg/update() ...
            tim/idleUpdate()
                  MsgToAph()

      rtt/stateSet(STA_CLOSING)
      rtt/stateSet(STA_MINI_WAIT)
      rtt/stateSet(STA_WAITING)
      rtt/stateSet(STA_EXITING)

      snprintf()
      MsgToAph()
      rt_get_adr()
      rt_mbx_send_if()
      epp/nextUpdate()
            epp/update() ...
      msg/update() ...
rtt/end()
      rtt/rttClose()
            rt_make_soft_real_time()
            stop_rt_timer()
            rt_get_adr()
            rt_task_delete()
            rt_mbx_delete()
      rtt/intClose()
            signal()
      printf()
      fprintf()
```

### 3.4. Firmware Design

The main function for the adaptable controllers (source in apc.c) initializes and runs all main functions. All controllers have the same program loaded, but with a unique ID [1 … 5] that is used to assign prioritized interrupt services and select the needed functions. ID 0 is reserved for broadcast messages (not used at this time).

Controller 1 is unique, since it is the only controller that has direct access to the real-time processor. It does the following:

It communicates with the host via parallel interface,
communicates with the other controllers via CAN,
checks panel keys, and
updates the display.

Controllers 2, 3, 4, and 5 are mainly used for the servo systems. They do the following:

Check and update position and input encoder counts,
adjust servo position,
check keys and switches,
check for messages from other controllers.

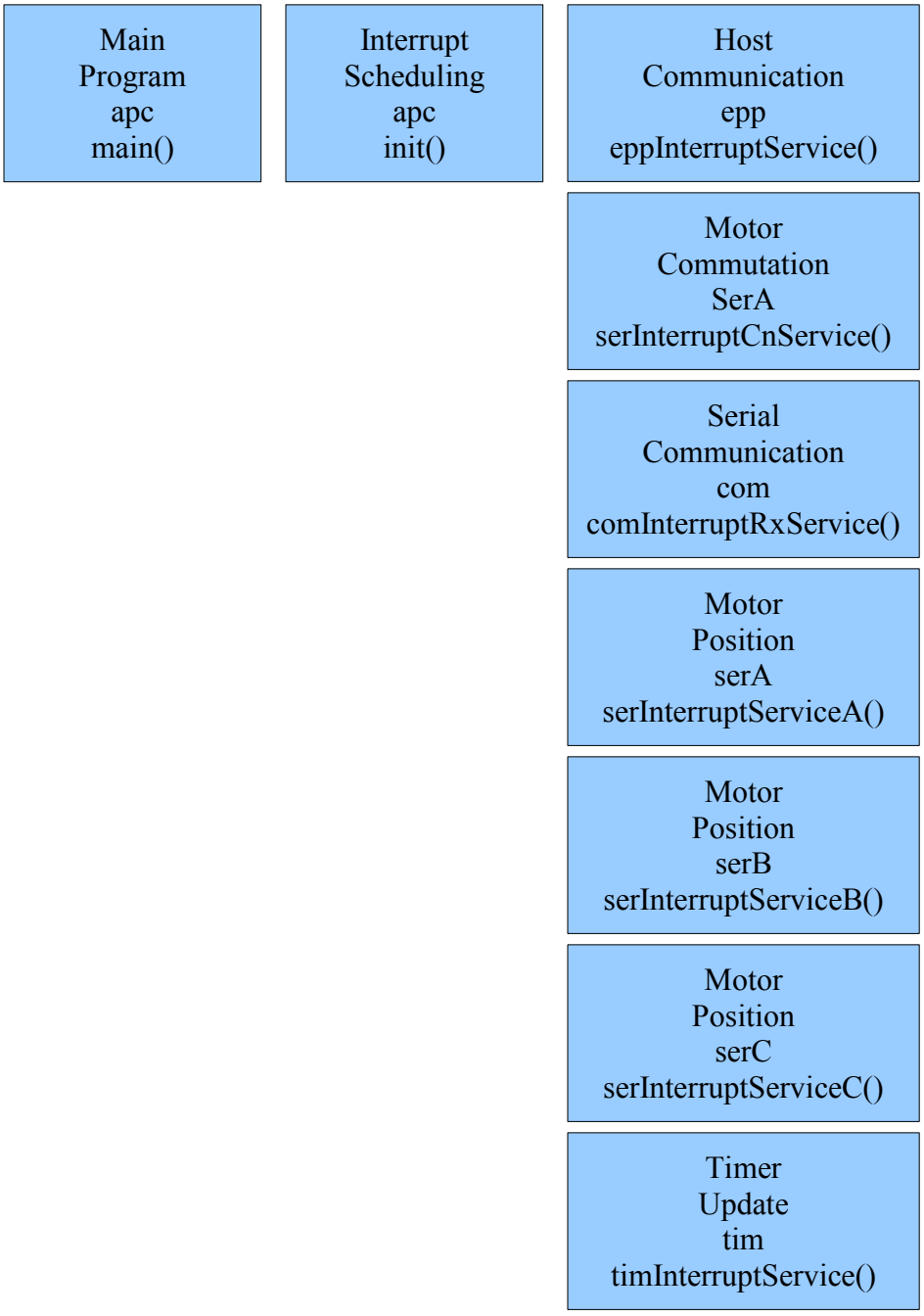
Interrupts suspend other modules depending on their priorities:

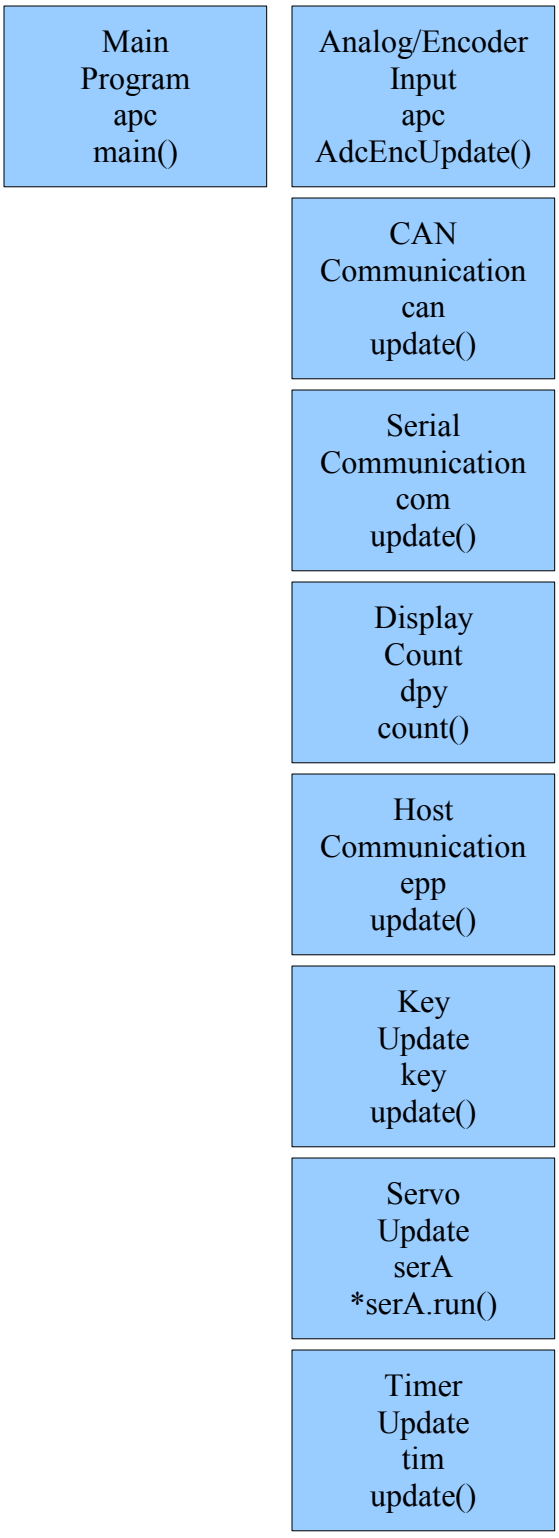
highest At the highest priority, communication modules with host (parallel interface), other external devices (serial interface etc.), or commutation for brush-less or stepper motors.

lower Servo module positions it's carriage. After updating the actual position, it adjusts the power to the motors to move the carriages toward the nominal position.

lower Timer module keeps track of time and updates counters for functions that need to be done periodically.

lowest At the lowest priority, a loop runs at maximum speed. Communication messages are interpreted and buffered. The display and other nominal settings are updated. If needed (indicated by the timer counters) the host is updated with the carriage's actual position, wheel movements, status and other information as well as key and light settings are checked and updated.






As indicated in the diagram, the main controller modules are apc, can, com, dpy, epp, key, serA, and tim. In addition, there are serB and serC modules to add more servo functionalities.

Since the program runs as an endless loop, the sequence of the main functions (as indicated) does not matter. In addition, the modules are only used if needed.

Firmware Flow:

```
apc/main()
      apc/adcInit()
      apc/encInit()
      apc/pwmInit()
      apc/refInit()
            apcGet()
            canGet()
            serAGet()
            serBGet()
            serCGet()
            timGet()
      apc/init()
      apc/stateSet(STA_INITIALIZING)
            can/init()
                  can/bufferClear()
            com/init()
            dpy/init()
                  dpy/update()
            epp/init()
            key/init()
            serA/init()
                  serOpen/serInit()
            serB/init()
                  serOpen/serInit()
            serC/init()
                  serOpen/serInit()
            tim/init()

      apc/stateSet(STA_CONFIGURING)
      apc/stateSet(STA_OPENING)
            serA/open()
                  serOpen/serConfig()
                        serSet/serTimeSet()
                              timGet()
                  serOpen/serDcConfig()
                  serTest/serTest()
                        serTest/serKineticTest()
                        serTest/serErrorSet()
                        serTest/serPidTest()
                        serTest/serLimitTest()
                        serTest/serIndexTest()
                        serSet/serPosTargetSet()
      apc/stateSet(STA_MINI_RUN)
      apc/stateSet(STA_RT_LOAD)
      apc/stateSet(STA_HARDWARE_LOAD)
      apc/stateSet(STA_INITIAL_LOAD)

      apc/stateSet(STA_RUNNING)
            can/update()
                  eppSet()
                  can/bufferRead()
                  canRead()
                        apcGet()
                        canGet()
                        comGet()
                        dpyGet()
                        eppGet()
                        keyGet()
                        serGetA()
                        serGetB()
```

```
                        serGetC()
                        timGet()
                        apcSet()
                        canSet()
                              eppSet()
                              can/bufferClear()
                              apcSet()
                        comSet()
                        dpySet()
                        eppSet()
                        keySet()
                        serSetA()
                        serSetB()
                        serSetC()
                        timSet()
                com/update()
                      canSet()
                dpy/count()
                      dpy/update()
                epp/update()
                      apcSet()
                      canRead()
                      canSet()
                key/update()
                      canSet()
                serA/update()
                tim/update()
                      timStream/timStreamGet()

          apc/stateSet(STA_LMT_COMPARATOR_RUNNING)
          apc/stateSet(STA_LMT_MODEL_RUNNING)
          apc/stateSet(STA_LMT_PROFILE_RUNNING)
          apc/stateSet(STA_P_COMPARATOR_RUNNING)
          apc/stateSet(STA_P_MODEL_RUNNING
          apc/stateSet(STA_P_PROFILE_RUNNING)
          apc/stateSet(STA_S9_COMPARATOR_RUNNING)
          apc/stateSet(STA_S9_MODEL_RUNNING)
          apc/stateSet(STA_S9_PROFILE_RUNNING)
          apc/stateSet(STA_IDLE)
          apc/stateSet(STA_CLOSING)
          apc/stateSet(STA_MINI_WAIT)
          apc/stateSet(STA_WAITING)
          apc/stateSet(STA_EXITING)


comInterruptRxService()
eppInterruptService()
serInterruptCnService()
serInterruptServiceA()
serInterruptServiceB()
serInterruptServiceC()
timInterruptService()
```

### 3.5. Software Usage and Configuration

The aph program can be started with several arguments. The syntax is:

aph [-c configFile] [-dD debugLevel] [-E debugLevel] [-f fifoSize] [-i] [-l logFile] [-m mode] [-pP parameterFile] [-s] [-t timerFile] [-u timeOut] [-v logFile] [-x flag]

The arguments are:

| | |
|---|---|
| -c configFile | Configuration file to be used. |
| -d debugLevel | Debug and log level; 0 (nothing) to 9 (anything) |
| -D debugLevel | Ignore errors and set debug and log level; 0 (nothing) to 9 (anything) |
| -E debugLevel | Write errors to standard error stream |
| -f fifoSize | Number of bytes available for message box |
| -i | Interactive user interface. |
| -l logFile | Append to logFile |
| -m mode | Initial mode after reset |
| -p parameterF | Parameter file to be used when starting and ending the program. |
| -s | Silent, no bell |
| -t timerFile | Timer file to be used |
| -u timeOut | Illumination timeout in minutes |
| -v logFile | Append to logFile all host communication. |
| -x flag | Flag value |

The rtt syntax is:

rtt [-a portAddress] [-c comTimeNs] [-deE] [-f fifoSize] [-iI timeMs] [-mM] [-p priority] [-rRs] [-t period] [-x flag] [-z]

The arguments are:

| | |
|---|---|
| -a portAddress | Paralle port address |
| -c comTimeNs | Minimum communicaiton time needed per update |
| -d | Send only different servo coordinates |
| -e | Log extended warnings and error messages |
| -E | Without warnings and error messages |
| -f fifoSize | Number of bytes available for message box |
| -i timeMs | Run initial tests and exit |
| -I timeMs | Start delay for tests |
| -m | Lock process into memory |
| -M | Swap process from memory |
| -p priority | Task priority |
| -r | Hard real-time task |
| -R | Soft real-time task |
| -s | Compute servo profile |
| -t periodMs | Task period in ms |
| -x flag | Flag value |
| -z | for lmt, model Z only mode, all input goes to Z |

The kbd syntax is:

kbd [-D] [-f fifoSize] [-x flag]

The arguments are

| | |
|---|---|
| -D | Diagnostics mode |
| -f fifoSize | Number of bytes available for message box |
| -x | Flag value |

The mou syntax is:

mou

The arguments are:

| | |
|---|---|
| -a factor | A (X) integer factor |
| -b factor | B (Y) integer factor |
| -c factor | C (Z) integer factor |
| -d factor | D (P) integer factor |
| -D | Diagnostics mode |
| -f fifoSize | Number of bytes available for message box |
| -k | No key events |
| -m | No move events |
| -n device | Device name |
| -p | Toggle input when pressing middle key: xy and pz |
| -t | Toggle input when pressing middle key xy and z |
| -x flag | Flag value |
| -0 | No x input |
| -1 | Use x as A input |
| -2 | Use x as B input |
| -3 | Use x as C input |
| -4 | Use x as D input |
| -5 | No y input |
| -6 | Use y as A input |
| -7 | Use y as B input |
| -8 | Use y as C input |
| -9 | Use y as D input |

The sas and sal syntax is

sas
sal

The arguments are:

| | |
|---|---|
| -f fifoSize | Number of bytes available for message box |
| -l file | Loop reading file |
| -m | Monitor mode |
| -r file | Read file |
| -s | None-interactive streaming mode |
| -t ms | Poll timeout in ms |
| -x flag | Flag value |

The sbc syntac is:

sbc

The arguments are:

| | |
|---|---|
| -D | Diagnostics mode |
| -f fifoSize | Number of bytes available for message box |
| -t ms | Poll timeout in ms |
| -x flag | Flag value |

The parameter and configuration files can be viewed and edited with a standard editor. Essentially, they can hold the same information. However, there are two main differences to be considered. First, since the same variable are affected, the reading sequence is important. When the program starts, internal defaults values are loaded, then the parameter file is read, and finally the configuration file is read.

Second, the program overwrites the parameter file to save the internal variables. Therefore, the parameter file should not be edited manually. However, it can serve as a useful reference to create a configuration file to enter system specific values. The syntax is:

VARIABLE value # Comment

| | |
|---|---|
| VARIABLE | is the name of the variable (see str.c) |
| value | is any numerical, character, or string value. Note that strings need to be enclosed in quotes. |
| # Comment | is any comment that is preceded by the # character. |

See appendix O for a listing and description of configuration options.

Each subsystem (keys, gui, ...) holds it's own configuration and variables.

Initialization:

| | |
|---|---|
| init() | is called to set the variables in the subsystems to default values. |
| cfg/cRead() | is called to read the file and pass variable names and values to the subsystems. If present, a parameter file (similar to the configuration file) is read and processed. |
| open() | is called to put the subsystem into running state. During the running state or when the system shuts down, values are saved to the parameter file via cfg/cWrite |
| update() | is called repeatably while the system is running |
| end() | is called when the system is shutting down |

Errors:
No error are indicated by return values of functions. Instead, global functions are used:

| | |
|---|---|
| aphErrorSet | is called to indicate an error in user space. |
| rttErrorSet | is called to indicate an error in the real-time system. |
| apcErrorSet | is called to indicate an error in the controller system. |

All errors are checked in user space. If recoverable, the subsystems will clear the errors.

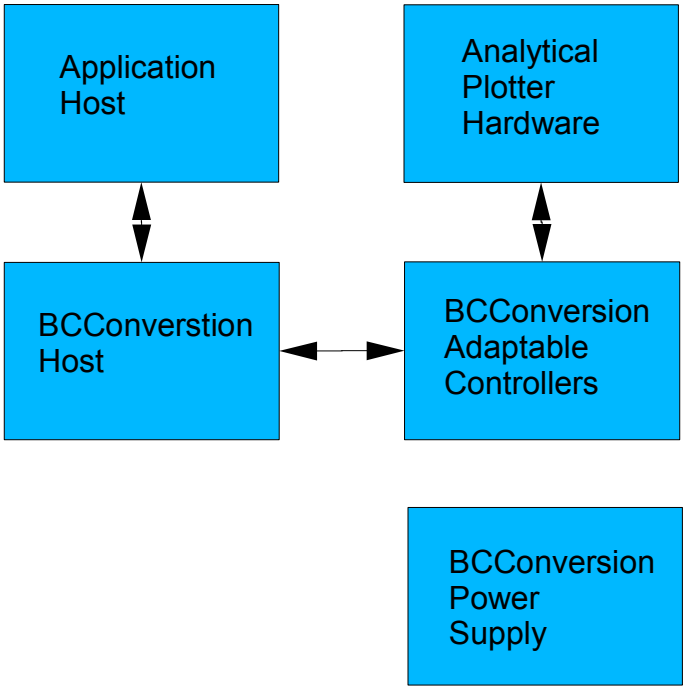
Function location:
External functions are in alphabetical order at the button of each subsystem file.

Comments:
Some compilers do not allow for nested comments. If possible use // as comments. Only if a big block needs to be commented out, use /* */.

Messages:

| | |
|---|---|
| can | are “CAN” messages that are implemented on the controllers. |
| msg | are used by aph and rtt to communicate. In addition, mou, kbd, sim and tou put messages into the aph mailbox and sip puts messages into rtt mailbox. |

### 3.6. Hardware Design




The BCConversion real-time host is a standard PC with a serial interface that is needed for an application host. A parallel port is needed to communicate with the first adaptable controller. A PC power supply provides the power for the BCConversion host, the controllers and the analytical plotter.

The adaptable controllers are a minimum of 5 identical circuits. The circuits can drive the left side, the right side, and additional circuits can be used for optional servo systems like zoom, dove, or other drives.

### 3.6.1. Connector Configurations

| Power connector P with key #6: | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | A<br>+48V | | B<br>+48V | | C<br>+48V | | D<br>+48V | |
| E<br>+24V | | F<br>+24V | | H | | J<br>+24V | | K<br>+24V |
| | L<br>+12V | | M<br>+12V | | N<br>+12V | | P<br>+12V | |
| R<br>+12V | | S<br>+12V | | T<br>+12V | | U<br>+12V | | V<br>+12V |
| | W<br>GND | | X<br>GND | | Y<br>GND | | Z<br>GND | |
| a<br>GND | | b<br>GND | | | | c<br>GND | | d<br>GND |
| | e<br>GND | | | | | | f<br>GND | |
| | h<br>GND | | | | | | j<br>GND | |
| k<br>GND | | l<br>GND | | | | m<br>GND | | n<br>GND |
| | p<br>GND | | r<br>GND | | s<br>GND | | t<br>GND | |
| u<br>+5V | | v<br>+5V | | w<br>+5V | | x<br>+5V | | y<br>+5V |
| | z<br>+5V | | AA<br>+5V | | BB<br>+5V | | CC<br>+5V | |
| DD<br>-12V | | EE<br>+3.3V | | FF<br>Reset | | HH<br>+3.3V | | JJ<br>-12V |
| | KK<br>Sec. Power | | LL | | MM | | NN<br>Buzzer | |

For AC10 connector configuration see the AC10 manual.

## 4. Implementation

For system requirements, environmental requirements, and hardware installation information see the BCConversion User Manual.

### 4.1. Software Installation

The software is pre-installed for new systems. However, if an update is needed or the system was repaired, below are procedures to install the software from scratch.

Requirements:

1. BCConversion hardware.
2. BCConversion software on CD and USB drive.
3. CD drive connected if software has to be copied to USB drive.
4. Keyboard, monitor and power cord connected to the host-processor.

#### 4.1.1. Jetway 625EMP800 Motherboard Setup

Procedure:

1. Press power button followed by [Delete] to enter the BIOS.
2. Select and “Load Standard Defaults”
3. In “Standard CMOS Features”, set the date and time, “Drive A” as “None”, and set “Halt On” to “All, But Keyboard”
4. In “Advanced BIOS Features”, set “First Boot Device” to “Floppy”, “Second” to “CDROM”, and “Third” to “HDD-0”
5. In “Advanced Chipset Features”, in “AGP Timing Settings”, set “AGP Aperture Size” to “4MB”, set “VGA Share Memory Size” to “16MB”
6. In “Integrated Peripherals”, in “OnChip Device Function” set “AC97 Audio Device” to “Disabled”.
7. In “Integrated Peripherals”, in “Onboard Super IO Function”, set “Onboard FDD Controller” to “Disabled”, and set “Onboard Serial Port 2” to “Disabled”.
8. In “Integrated Peripherals”, set “Init Display First” to “Onboard VGA”.
9. In “Power Management Setup”, set “State after Power Failure” to “Always On”
10. “Save & Exit Setup”

#### 4.1.2. Jetway 626EMP1G Motherboard Setup

Procedure:

1. Press power button followed by [Delete] to enter the BIOS.

2. Select and “Load Standard Defaults”

3. In “Standard CMOS Features”, set the date and time, “Drive A” as “None”, and set “Halt On” to “All, But Disk/ Key”

4. In “Advanced BIOS Features”, set “First Boot Device” to “USB-HDD”, “Second” to “CDROM”, and “Third” to “HDD-0”

5. In “Advanced Chipset Features”, in “AGP Timing Settings”, set “AGP Aperture Size” to “4MB”, set “VGA Share Memory Size” to “16MB”

6. In “Integrated Peripherals”, in “OnChip Device Function” set “Game Port Address” to “Disabled” and “USB Device Legacy Support” to “Always On”.

7. In “Integrated Peripherals”, in “Onboard Super IO Function”, set “Onboard FDD Controller” to “Disabled”, and set “Onboard Serial Port 2” to “Disabled”.

8. In “Integrated Peripherals”, set “Init Display First” to “Onboard VGA”.

9. In “Power Management Setup”, set “State after Power Failure” to “Always On”

10. “Save & Exit Setup”

#### 4.1.3. Intel D201GLY Motherboard Setup

Procedure:

1. Press power button followed by [F2] to enter the BIOS.

2. [F9] to load Setup Defaults

3. In “Main”, set the date and time

4. In “Power”, set “After Power Failure” to “Power on”

5. In “Boot”, set “Boot USB Devices First” to “enable”

6. [F10] to “Save and Exit”

#### 4.2.1. Linux Software Installation on Hard Disk Drive

After the motherboard setup, the system boots from CD. To install the software on a new disk dive, do the following:

| | |
|---|---|
| 1. [Enter] | To use GMT as timezone |
| 2. [Enter] | To configure the system for local clock |
| 3. [Enter] | To use the default local setting |
| 4. [Enter] | To accept the settings |
| 5. [Enter] | To activate the console |
| 6. fdisk -l | To check device name, boot flag is set, ID is set to “b” W95 FAT32 |
| 7. mkdosfs -F 32 /dev/hda1 | If needed, format the drive |
| 8. mount /dev/hda1 /mnt | Mount the drive |
| 9. mkdir cd | Make mount point for cd |
| 10. mount /dev/cdrom cd | Mount cd |
| 11. cp -r cd/aph-* /mnt | Copy content of aph* directrory to the drive |
| 12. umount cd | Unmount cd |
| 13. cd /mnt/boot/syslinux | Change directory |
| 14. cat mbr.bin > /dev/hda | Install master boot record on drive |
| 15. ./syslinux -d boot/syslinux /dev/hda1 | Install boot loader |
| 16. cd | Release /mnt |
| 17. umount /mnt | Unmount the drive |
| 18. halt | |

Remove CD drive keyboard and monitor. Install covers. Connect USB drive. Press power button to test, calibrate and use the system.

#### 4.2.2. Linux Software Installation on USB Drive

After the motherboard setup, the system boots from CD. To install the software on a new USB dive, do the following:

| | |
|---|---|
| 1. [Enter] | To use GMT as timezone |
| 2. [Enter] | To configure the system for local clock |
| 3. [Enter] | To use the default local setting |
| 4. [Enter] | To accept the settings |
| 5. [Enter] | To activate the console |
| 6. fdisk -l | To check name, boot flag is set, ID is set to “e” W95 FAT16 (LBA) |
| 7. mkdosfs /dev/sda1 | If needed, format the drive |
| 8. mount /dev/sda1 /mnt | Mount the drive |
| 9. mkdir cd | Make mount point for cd |
| 10. mount /dev/cdrom cd | Mount cd |
| 11. cp -r cd/aph-* /mnt | Copy content of aph* directrory to the drive |
| 12. umount cd | Unmount cd |
| 13. cd /mnt/boot/syslinux | Change directory |
| 14. cat mbr.bin > /dev/sda | Install master boot record on drive |
| 15. ./syslinux -d boot/syslinux /dev/sda1 | Install boot loader |
| 16. cd | Rrelease /mnt |
| 17. umount /mnt | Unmount the drive |
| 18. halt | |

Remove CD drive keyboard and monitor. Install covers. Press power button to test, calibrate and use the system.

#### 4.2.3. Windows Software Installation on USB Drive

To install the software on a USB dive from the CD within Microsoft Windows, do the following:

| | |
|---|---|
| 1. [My Computer] | Find drive names. In this example d: for CD, e: for USB drive |
| 2. [Start][Run...][cmd] | Get command window |
| 2. format e: | If needed, delete content and format USB drive |
| 3. cd /d d:\aph* | Change to CD's aph directory |
| 4. xcopy /s *.* e: | Copy content to USB drive |
| 5. cd /d e:\boot\syslinux | Change to USB's syslinux directory |
| 6. syslinux -am -d boot/syslinux e: | Install boot loader |
| 7. exit | Exit command window |
| 8. [Savely Remove Hardware][e:] | Remove USB drive from system |

Connect USB drive to plotter electronics. Install covers. Press power button to test, calibrate and use the system.

### 4.3. Firmware Installation

The firmware is typically pre-installed. However, if an update is needed or the system was repaired, below are the steps to install the software from scratch.

Requirements:

1. Microchip ICD2 Programmer connected to a PC.

2. AC10 Programming cable connected to power, ICD2 and the controller.

3. All software on a PC.

Procedure:

1. Start Microchip's MPLAB IDE

2. Project -> Open -> apConvertion.mcp

3. File -> Import -> apConvertion.hex

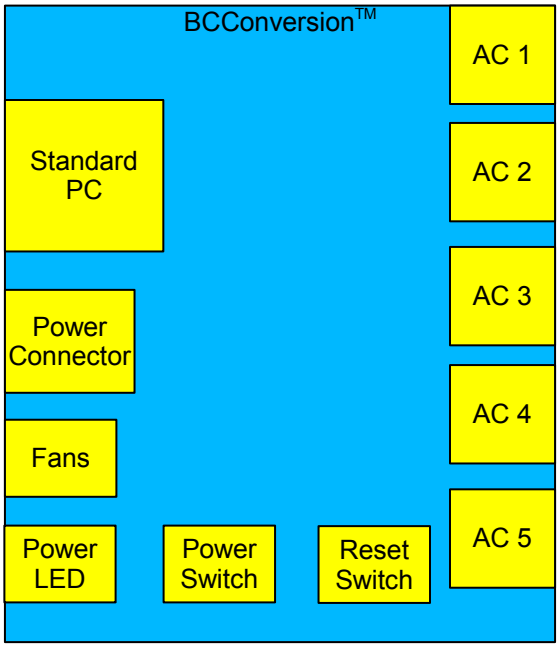
4. Programmer -> Connect (note, “Target Device dsPIC30F6010 found” in the output window)

5. Programmer -> Program

6. Repeat the last two steps for all controllers

### 4.4. Hardware

Below is an outline of the complete hardware. A standard PC (power supply, motherboard with CPU, RAM, video, Ethernet, and USB drive), 5 adaptable controllers, power connector, power LED, power switch, and reset switch are in the electronics enclosure along with two fans in the cover of the enclosure. PC, power connector, LED, power switch and fans are all connected within the enclosure. However, the reset switch (used for controller reset) is only wired to the power connector and nothing is connected to any controller on the inside.




The following plugs connect the hardware with the external world:
From the PC:

- a keyboard connector (not needed)
- a mouse connector (not needed)
- one Ethernet connector (not needed)
- two USB connectors (needed for software configuration and updates)
- a parallel connector (needed for communication with first controller)
- a serial RS232 connector (COM1 needed for communication with application host)
- a video connector (not needed)
- game and audio connectors (not needed)

For details on the power connector see 3.6.1.

For details on the controller 56 pin MALCO/ELCO/EDAC/Connectral connectors, see AC10 manual.

For details on locating and connecting cables, see installation manual.

**5. Updating, Testing and Saving**

To re-compile the program do this:

| | |
|---|---|
| mount -r /dev/sda1 /mnt | Mount drive |
| cd /usr/src | Change directory |
| tar -xjvf /mnt/data/aph* | Extract aph files |
| umount /mnt | Release drive |
| cp /usr/realtime/lib/* /lib | Copy library files |
| cd /tmp | Change directory |
| /usr/src/aph*/configure | Check and configure |
| make install-strip | Compile and install software |

To start and test each file interactively:

| | |
|---|---|
| start -i | Load real-time modules |
| rtt | Start real-time program |
| aph -i -d 8 -l /tmp/log | Start aph program in interactive diagnostics mode |
| mou -D -n /dev/input/mouse0 | Start mouse program in diagnostics mode |
| kbd < tty1 | Start keyboard |
| stop | Stop the programs |

To start in default mode:

| | |
|---|---|
| mount -r /dev/sda1 /a | Mount drive |
| start | Start programs in default mode |
| stop | Stop the programs |

To save updated version:

| | |
|---|---|
| cd /usr/src | Change directory |
| mount /dev/sda1 /mnt | Mount drive |
| tar -cjvf /mnt/data/aph-newVersion aph* | Compress and save directory to the aph-newVersion file on the drive |
| umount /mnt | Release drive |

**Appendix A; Copy of GNU's General Public License**

## GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

## Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

## TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

**0.** This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

**1.** You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

**2.** You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- **a)** You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- **b)** You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- **c)** If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

**3.** You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- **a)** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- **b)** Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- **c)** Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

**4.** You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

**5.** You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

**6.** Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

**7.** If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

**8.** If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

**9.** The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

**10.** If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

**NO WARRANTY**

**11.** BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD

THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

**12.** IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

**Appendix D; CD Directory and File Names**

| | |
|---|---|
| /aph-v-n | Directory with binary aph software, v indicates version, n indicates serial number |
| /boot | Directory with data needed to boot from the CD |
| /data | Directory with compressed data files needed to build the system |
| /src | Directory with documentation and source files for AC10 and aph |
| Copying.txt | Copy of GNU GENERAL PUBLIC LICENSE |
| Copyright.txt | Copyright information |
| Readme.txt | Latest information |
| index.html | Index file to explore the CD with a web browser |

| | |
|---|---|
| /aph-v-n | |
| ./boot | Directory with data needed to boot from flash or hard drive |
| Copying.txt | Copy of GNU GENERAL PUBLIC LICENSE |
| Copyright.txt | Copyright information |
| Readme.txt | Latest information |
| aph_sn | Serial number of the system |
| aphConfig | System configuration |
| auto.sh | Startup script |

| | |
|---|---|
| /aph-v-n/boot | |
| ./syslinux | Directory with syslinux files |
| aclogo.lss | Adaptable Controllers logo |
| aph | Linux kernel |
| aphrd.gz | Linux compressed ram file |
| menu | First boot menu |

| | |
|---|---|
| /aph-v-n/boot/syslinux | |
| README | Copyright information for syslinux |
| mbr.bin | Master boot record |
| menu.c32 | Second boot menu program |
| syslinux | Linux version for syslinux |
| syslinux.cfg | Configuration for second menu |
| syslinux.exe | Windows version of syslinux |

| | |
|---|---|
| /boot | |
| ./isolinux | Directory with isolinux files to boot from CD |
| acdev | Linux kernel |
| acdevrd.gz | Linux compressed ram file |
| aclogo.lss | Adaptable Controllers logo |
| menu | First boot menu |

| | |
|---|---|
| /boot/isolinux | |
| README | Copyright information for isolinux |
| isolinux.bin | CD boot program |
| isolinux.cfg | Configuration for second menu |
| menu.c32 | Second boot menu program |

| | |
|---|---|
| /data | |
| ac10-v.tar.bz2 | AC10 documentation and sources |
| aph-v.tar.bz2 | aph documentation and sources |
| busybox-v.tar.bz2 | busybox sources |
| cdrtools-v.tar.bz2 | cdrecord sources |
| linux-v.tar.bz2 | Linux kernel sources |
| rtai-v.tar.bz2 | rtai sources |
| syslinux-v.tar.bz2 | syslinux sources |
| zisofs-tool-v.tar.bz2 | ISO tool sources |

/src

| | |
|---|---|
| ./ac10-v | AC10 documentation and source files, v indicates version |
| ./aph-v | aph documentation and source files, v indicates version |

/src/ac10-v

| | |
|---|---|
| ./doc | Directory with documentation |
| ./src | Directory with source files |
| Copying | Copy of GNU GENERAL PUBLIC LICENSE |
| Copyright | Copyright information |
| Credits | Credit information |
| Readme | Latest information |
| Version | Version information |

/src/ac10-v/doc

| | |
|---|---|
| src | Source code documentation |
| AC10Manual.pdf | Hardware documentation |

/src/ac10-v/src/firmware

| | |
|---|---|
| bldc.c | Brush-less DC motor test program |
| can.c | CAN network test program |
| dimmer.c | Dimmer test program |
| epp.c | Parallel port test program |
| led.c | LED test program |
| stepper.c | Stepper motor test program |
| *.gld | Linker files |
| *.mcp | Microchip IDE project files |

/src/aph-v

| | |
|---|---|
| ./config | Directory with default and example configuration files |
| ./doc | Directory with documentation |
| ./images | Directory with images |
| ./scripts | Directory with scripts |
| ./src | Directory with source code |
| AUTORS | Credit file |
| ChangeLog | |
| COPYING | Copy of GNU GENERAL PUBLIC LICENSE |
| COPYRIGHT | COPYRIGHT information |
| INSTALL | Source compilation and installation information |
| Makefile.* | autoconf file |
| NEWS | |
| README | Latest information |
| aclocal.m4 | autoconf file |
| configure | autoconf output file |
| configure.ac | autoconf input file |
| index.html | Index file to explore the aph information with a web browser |

/src/aph-v/config

| | |
|---|---|
| ./autoconf | autoconf directory |
| aphConfig | default configuration |
| aphConfig* | Example configuration files |
| aphParam | Parameter file for configurations |
| inputkey.def | Default LMT key definition |
| inputkey.* | LMT key definition template files |
| stagesys.dat | LMT stage definition file |
| zeropnt.dat | LMT stage center file |

/src/aph-v/doc

| | |
|---|---|
| ./src | Directory with source documentation |
| BCConversionInfo.pdf | General information |
| BCConversionInstallation.pdf | Installation information |
| BCConversionTechnical.pdf | Technical information |
| BCConversionUser.pdf | User information |

/src/aph-v/doc/src

| | |
|---|---|
| ./html | Directory with source documentation for browser |
| Doxyfile | Configuration file for source documentation |

/src/aph-v/images

| | |
|---|---|
| AC-logo.gif | Adaptable Controllers logo |

/src/aph-v/scripts

| | |
|---|---|
| lmt | LMT configuration script |
| start | start and stop script |

/src/aph-v/src

| | |
|---|---|
| ./aph | Directory with aph thread source files |
| ./firmware | Directory with firmware source files |
| ./include | Directory with header files |
| ./kbd | Directory with keyboard thread source files |
| ./mou | Directory with mouse thread source files |
| ./mplab | Directory with Microchip IDE files |
| ./rtt | Directory with real-time thread source files |
| ./sim | Directory with host simulation source files |
| ./sip | Directory with plotter simulation source files |
| ./tes | Directory with test files |
| ./tou | Directory with touch screen source files |
| Makefile.* | autoconf files |

/src/aph-v/src/aph

| | |
|---|---|
| Makefile.* | autoconf files |
| apc.c | Adaptable plotter controller module |
| aph.c | Main function and host module |
| apiLMT.c | Application interface module for LMT protocol |
| apiP.c | Application interface module for P protocol |
| apiS9.c | Application interface module for S9 protocol |
| can.c | Controller Area Network (CAN) communication module |
| cfg.c | Configuration module |
| com.c | Serial communication module |
| dpy.c | Display and LED module |
| epp.c | Enhanced parallel port module |
| grd.c | Ground module |
| gui*.c | Graphical user interface (GUI) modules |
| inp.c | Operator input module |
| key.c | Key and button module |
| lmt.c | Module for LMT mode |
| log.c | Log module |
| msg.c | Message module |
| phc.c | Photogrammetric module for comparator mode |
| phm.c | Photogrammetric module for model mode |
| pro.c | Photogrammetric module for profile mode |
| rec.c | Record module |
| rtt.c | Real-time thread setup module |

| | |
|---|---|
| ser.c | Servo module |
| str.c | String module |
| tes.c | Test module |
| tim.c | Timer module |

/src/aph-v/src/firmware

| | |
|---|---|
| apc.c | Main function and control module |
| can.c | Controller Area Network (CAN) communication module |
| com.c | Serial communication module |
| dpy.c | Display and LED module |
| epp.c | Enhanced parallel port module |
| key.c | Key and button module |
| ser*.c | Servo module |
| tim*.c | Timer module |

/src/aph-v/src/kbd

| | |
|---|---|
| Makefile.* | autoconf files |
| kbd.c | Keyboard thread source code |

/src/aph-v/src/mou

| | |
|---|---|
| Makefile.* | autoconf files |
| mou.c | Mouse thread source code |

/src/aph-v/src/mplab

| | |
|---|---|
| apConvertion.cod | Symbol and debug file generated by the assembler or linker |
| apConvertion.cof | Debug file for MPLAB |
| apConvertion.hex | Hex 32 file with binary code for programming the controllers |
| apConvertion.lst | Listing of binary and source code information generated by the assembler |
| apConvertion.map | Listing of memory layout generated by the linker |
| apConvertion.mcp | Project information generated by MPLAB |
| apConvertion.mcs | Saved Workspace information generated by MPLAB |
| apConvertion.mcw | Workspace information generated by MPLAB |
| p30f6010.gld | Linker file for dsPIC30F6010 |
| p30f6010A.gld | Linker file for dsPIC30F6010A |

/src/aph-v/src/rtt

| | |
|---|---|
| Makefile.* | autoconf files |
| can.c | Controller Area Network (CAN) communication module |
| epp.c | Enhanced parallel port module |
| inp.c | Operator input module |
| lmt.c | Module for LMT mode |
| msg.c | Message module |
| phc.c | Photogrammetric module for comparator mode |
| phm.c | Photogrammetric module for model mode |
| pro.c | Photogrammetric module for profile mode |
| rtt.c | Main function and real-time thread module |
| ser.c | Servo module |
| tim.c | Timer module |

/src/aph-v/src/sim

| | |
|---|---|
| Makefile.* | autoconf files |
| arg.c | Argument functions |
| dat.c | Communication data functions |

| | |
|---|---|
| gui.c | User interface functions |
| inp.c | File input functions |
| int.c | Interrupt functions |
| key.c | Keyboard input functions |
| msg.c | Message functions |
| rtt.c | Real-time thread functions |
| sal.c | Main LMT simulation and monitoring functions |
| sas.c | Main S9 simulation and monitoring functions |

/src/aph-v/src/sip

| | |
|---|---|
| Makefile.* | autoconf files |
| apc.c | Simulation of apc firmware functions |
| can.c | Simulation of can firmware functions |
| com.c | Simulation of com firmware functions |
| dpy.c | Simulation of dpy firmware functions |
| epp.c | Simulation of epp firmware functions |
| key.c | Simulation of key firmware functions |
| sbc.c | Simulation of BC functions |
| ser.c | Simulation of ser firmware functions |
| tim.c | Simulation of tim firmware functions |

/src/aph-v/src/tes

| | |
|---|---|
| Makefile.* | autoconf files |
| beep.c | Buzzer test |
| display.c | Display functions for rtai latency test |
| dos2u.c | Carriage return removal |
| epp*.c | Parallel port test functions |
| kb.c | Keyboard test functions |
| latency.c | rtai latency test |
| mouse*.c | Mouse test functions |
| sal* | LMT simulation test files |
| sas* | S9 simulation test files |

/src/aph-v/src/tou

| | |
|---|---|
| Makefile.* | autoconf files |
| tou.c | Touch screen thread source code |

**Appendix H; Document History**

2.0
Adaptable Controllers; November 2008; Updated to CD installation for aph version 2.1

0.0.24
Adaptable Controllers; July 2007; Updated Motherboard and Debian 4.0 (LINUX 2.6.18) installation instructions

0.0.23
Adaptable Controllers; February 2006; Updated RTAI 3.3 and LINUX 2.6.15 installation instructions

0.0.21
Adaptable Controllers; April 2005; Cleanup for product sale; Removed old hardware information. Added AC10 information.

0.0.12
RenaiSys; November, 2003; Added GAP3 hardware schematics, part lists, ...

0.0.10
RenaiSys; March 10, 2003; Added implementation information; GAP2 hardware schematics, part lists, ...

0.0.06
RenaiSys; September 10, 2002; Switched design from passive counter and pulse-width modulator to Peripheral Interface Controller (PIC) microprocessors.

0.0.05
RenaiSys; June 10, 2002; Added design information

0.0.04
RenaiSys; April 10, 2002; Added BC and AC requirements as well as GAP1 hardware schematics

0.0.03
RenaiSys; March 10, 2002; Added S9 application interface, IMA requirements, and updated promotion drafts

0.0.02
RenaiSys; February 1, 2002; Updated draft

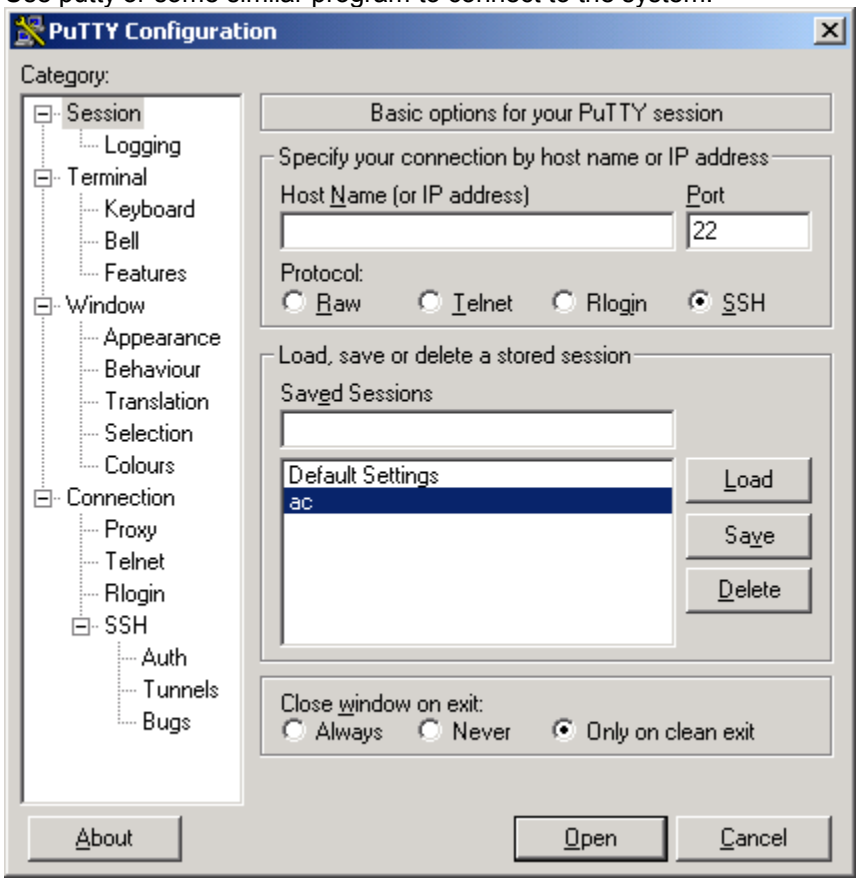
0.0.01
RenaiSys; December 12, 2001; Original

**Appendix N; Networking**

To connect to the system via network, make to following adjustments.

| | |
|---|---|
| cd /etc | Change directory |
| vi passwd | Add aph account |
| vi shadow | Remove password for root |
| passwd | Add root password |
| su aph | Change to aph |
| passwd | Add aph passwd |
| cd rc.d/init.d | Change directory |
| ./sshd start | Start sshd server |

Use putty or some similar program to connect to the system.



**Appendix P; Firmware Programming Cable**

| 56pin Elco Plug; Code 6-6; for Power | | |
|---|---|---|
| Pin | Color | Description |
| f | black | Ground |
| CC | red | +5V |

| 56pin Elco Plug; Code 1-1; for AC10 | | |
|---|---|---|
| Pin | Color | Description |
| A | blue | Data |
| B | green | Clock |
| h | red | +5V |
| j | black | Ground |
| y | orange | Reset |

| 6pin RJ11 Plug; for ICD2 | | |
|---|---|---|
| Pin | Color | Description |
| 1 | orange | Reset |
| 2 | red | +5V |
| 3 | black | Ground |
| 4 | blue | Data |
| 5 | green | Clock |
| 6 | | open |

**Appendix R; Resources**
Below are some on-line resources and books.

| | |
|---|---|
| Software resources: | |
| GNU General Public License, Free Software Directory, and more | http://www.gnu.org |
| Kernel Source and more | http://www.kernel.org |
| Kernel mailing list | http://www.tux.org/lkml |
| Real Time Linux from the New Mexico Institute of Technology | http://www.rtlinux.org |
| Real Time Linux Application Interface from the Politecnico di Milano | http://www.rtai.org |
| Open Source Development | http://sourceforge.net |
| Open Source Software | http://freshmeat.net |
| Online manual for Linux | http://tldp.org<br>http://linux.ctyme.com |
| Realtime Embedded Guide | http://people.mech.kuleuven.ac.be/~bruyninc/rthowto |
| Online manual for C++ | http://www.cplusplus.com/doc/tutorial |
| Device Driver Manual | http://www.xml.com/ldd/chapter/book |
| Kernel programming | http://www.kernelnewbies.org |
| Sun Microsystems' home for JavaDoc | http://java.sun.com/j2se/javadoc |
| OpenOffice.org Documentation Software | http://www.openoffice.org |
| Enhanced Machine Controller from the National Institute of Standards and Technology | http://www.isd.mel.nist.gov/projects/emc |
| EMC web site | http://www.linuxcnc.org<br>http://sourceforge.net/projects/emc |
| Open robot control software | http://www.orocos.org |
| Matrix arithmetic | http://www.euclideanspace.com/maths/algebra/matrix/arithmetic/index.htm |
| | |
| Electrical engineering resources: | |
| Electrical engineering | http://www.web-ee.com |
| Electronics design network | http://www.edtn.com |
| Electronics design magazine | http://www.e-insite.net/ednmag |
| Lessons in electric circuits by Tony R. Kubhaldt | http://www.ibiblio.org/obp/electricCircuits |
| Parallel Port port, EPP mode, IEEE 1284 | http://www.fapo.com/eppmode.htm |
| Intel hex file scpecifications | ftp://download.intel.com/support/processors/i960/devtools/INTELHEX.PDF |
| Schematics editor | http://www.bartels.de |
| Printed circuit board development software | http://www.cadsoft.de |
| Gerber data viewer | http://www.pentalogix.com |
| | |

| | |
|---|---|
| PIC microcontrollers | http://www.microchip.com |
| PIC resources | http://www.gnupic.org |
| PIC programmer by Byron Jeff | http://www.finitesite.com/d3jsys |
| | |
| Power design resources | http://www.powerdesigners.com/InfoWeb |
| Power electronics news | http://www.pcim.com |
| Simple pulse width modulation circuit | http://www.solorb.com/elect/solarcirc/pwm1 |
| Computer operated telescope with servo motors and stepper motors | http://www.bbastrodesigns.com/cot/cot.html |
| Computer controlled circuits | http://www.boondog.com |
| PC base motion controllers | http://www.robotics.com |
| PC base motion controllers | http://www.servotogo.com |
| Motion controller cards | http://www.mesanet.com/motioncardinfo.html |
| Precision Motion Controller from National Semiconductor | http://www.national.com/pf/LM/LM629.html |
| Motion Controller from J R Kerr | http://www.jrkerr.com/ |
| Microcontrollers from Intel | http://www.intel.com/design/embcontrol/ |
| Microcontrollers from Mircochip | http://www.microchip.com/ |
| Object oriented programmable circuit | http://www.oopic.com |
| Embedded system programming | http://ee.cleversoul.com |
| Chip directory | http://www.embeddedlinks.com/chipdir |
| | |
| Electronics component sources: | |
| Custom PCBs | http://www.pcbexpress.com |
| Dream, custom electronics | http://www.kdream.com |
| Jameco, common electronics parts | http://www.jameco.com |
| Digi-key, electronics parts | http://www.digikey.com |
| Allied electronics, electronics parts,<br>HCTL 1100 motion controller, HCTL2000 counter | http://www.alliedelec.com |
| Mouser, electronics parts | http://www.mouser.com |
| Alltonics, electronics and computer parts | http://www.alltronics.com |
| JDR, computer parts | http://www.jdr.com |
| Wirz, motion controllers and amplifiers | http://wirz.com |
| US Digital, encoders and counters | http://www.usdigital.com |
| DRC encoders | http://www.drc.com/encoder/intro.htm |
| Cleveland Motion Controls and encoders | http://www.cmccontrols.com |
| RSF encoders | http://www.rsf.net |
| Heidenhain encoders | http://www.heidenhain.com/product.html |
| | |
| Stock Drive Products, bearings and belts | http://www.sdp-si.com |

| | |
|---|---|
| | |
| Hardware datasheets: | |
| General directory for 74LS... and others, check the electrical engineering sites as well. | http://www.hanssummers.com/electronics/datasheets/ |
| Texas Instruments | http://www.ti.com |
| MM74C922 key encoder circuit information from the University of Illinois | http://www.ece.uiuc.edu/eshop/availablemodules/CPLD/mm74c922.pdf |
| ICM7218 display decoder circuit information from the Dartmouth Collage | http://engineering.dartmouth.edu/~engs031/databook/icm7218.pdf |
| LS7266 from US Digital | http://www.usdigital.com/pdf/LS7266R1%20Data%20Sheet.pdf |
| 8254 from Intel | http://developer.intel.com/design/periphrl/datashts/23124406.pdf |
| LMD18200 from National Semiconductor | http://www.national.com/ds/LM/LMD18200.pdf |
| | |
| 74LS45 | http://www-s.ti.com/sc/ds/sn7445.pdf |
| 74LS156 | http://www-s.ti.com/sc/ds/sn74156.pdf |
| | |
| Photogrammetry: | |
| International | http://www.isprs.org |
| National | http://www.asprs.org<br>http://www.rspsoc.org<br>http://www.auslig.gov.au<br>http://www.ign.fr<br>http://foto.hut.fi/seura/fsprs_eng.html<br>http://www.geomatics.kth.se/~ssff/index_e.html |

Books:
Systems Analysis & Design Methods by Jeffrey L Whitten, Lonnie D Bentley, Vicotr M Barlow; published by Irwin, Burr Ridge, Illinois; ISBN 0-256-09360-1

Elements of Photogrammetry by Paul R Wolf, Bon A Dewitt; published by Mc Graw Hill, Burr Ridge, Illinois; ISBN 0-07-292454-3

**Appendix Z; Credits**

This is at least a partial credits-file list of people that have contributed to the BCConversion. It is sorted by name and formatted to allow easy grepping and beautification by scripts. The fields are: name (N), email (E), web-address (W), PGP key ID and fingerprint (P), description (D), and snail-mail address (S).

Please write to the current coordinator if the list needs to be updated.

Thanks,
Rene

---

N: Rene Sonderegger
E: Rene.Sonderegger@RenaiSys.com
W: http://www.renaisys.com
D: Coordinator
S: 17212 Niwot Place
S: Parker, CO 80134
S: USA